1. What is statistics, and why is it important?

- Statistics is a branch of mathematics that deals with collecting, organizing, analyzing, interpreting, and presenting data. It helps us make sense of data and draw conclusions or make decisions based on it.

Why is Statistics Important-

1. Informed Decision-Making:
    - Businesses, governments, and individuals use statistics to make data-driven decisions (e.g., customer behavior, election predictions).
2. Problem-Solving:
    - Helps in identifying patterns, trends, and relationships in data that can solve real-world problems.
3. Research & Development:

- Essential in scientific research for analyzing experiments and validating results.

4. Quality Control:

- Used in manufacturing to maintain product quality by monitoring variations.

5. Forecasting:

- Helps predict future trends in finance, weather, healthcare, etc.

6. Data-Driven World:
    - In today's digital age, data is everywhere. Statistics is key to understanding and utilizing this data effectively.

2. What are the two main types of statistics?

- The two main types of statistics are:

1. Descriptive Statistics

- Descriptive statistics are used to summarize and describe the main features of a dataset.

2. Inferential Statistics

- Inferential statistics are used to make predictions or inferences about a population based on a sample of data.

3. What are descriptive statistics?

- Descriptive Statistics is a branch of statistics that involves organizing, summarizing, and presenting data in an easy-to-understand way. It helps us quickly grasp the key features of a dataset without making predictions or generalizations.

4. What is inferential statistics?

- Inferential Statistics is a branch of statistics that allows us to draw conclusions, make predictions, or test hypotheses about a larger population based on a sample of data.

5. What is sampling in statistics?

- Sampling is the process of selecting a subset (sample) from a larger group (population) to analyze and draw conclusions about the whole group.

6. What are the different types of sampling methods?

- ✅ 1. Probability Sampling:-
- Every member of the population has a known and equal chance of being selected.

a). Simple Random Sampling

- Every individual has an equal chance.
- Example: Drawing names from a hat.

b). Systematic Sampling

- Select every kth item from a list.
- Example: Every 10th person on a class roll.

c). Stratified Sampling

- Divide the population into groups (strata) and randomly sample from each.
- Example: Sampling students from each year (1st year, 2nd year, etc.).

d). Cluster Sampling

- Divide the population into clusters, randomly select some clusters, and include all members in them.
- Example: Choosing 3 colleges randomly and surveying all students from each.

❌ 2. Non-Probability Sampling:-

- Not all members have a known or equal chance of being selected. Useful for quick or exploratory research.

a). Convenience Sampling

- Choose whoever is easiest to reach.
- Example: Asking friends or classmates nearby.

b). Judgmental/Purposive Sampling

- Researcher selects based on their judgment.
- Example: Selecting only top-performing students for a skill survey.

c). Snowball Sampling

- Existing participants recruit new ones.
- Example: Used in studies involving rare populations like influencers or startup founders.

d). Quota Sampling

- Set quotas to ensure representation of specific characteristics.
- Example: 50% male and 50% female respondents in a survey.

7. What is the difference between random and non-random sampling?

- 1.Random Sampling (Probability

Sampling):- Definition:

    - Random sampling is a technique in which every member of the population has a known, non-zero, and equal chance of being selected.

Key Features:

- Based on the principle of probability.
- Reduces sampling bias.
- Ensures that the sample is representative of the population.
- Results are more reliable and generalizable.

Examples:

- Simple Random Sampling
- Systematic Sampling
- Stratified Sampling
- Cluster Sampling

2. Non-Random Sampling (Non-Probability

Sampling):- Definition:

- Non-random sampling is a method where not all members of the population have a known or equal chance of being selected.

Key Features:

- Selection is based on convenience, judgment, or other non-probabilistic factors.
- Higher risk of bias.
- Less generalizable to the entire population.
- Useful for exploratory research or when probability sampling is not feasible.

Examples:

- Convenience Sampling
- Judgmental or Purposive Sampling
- Snowball Sampling
- Quota Sampling

8. Define and give examples of qualitative and quantitative data.

- ✅ 1. Qualitative Data (Categorical

Data) Definition:

  - Qualitative data refers to non-numeric information that describes qualities or characteristics. It is used to categorize or label attributes of a population.

Examples:

- Gender (Male, Female, Other)
- Eye color (Brown, Blue, Green)
- Blood type (A, B, AB, O)
- Education level (High School, Bachelor's, Master's)
- Customer feedback (Satisfied, Neutral, Dissatisfied)

✅ 2. Quantitative Data (Numerical

Data) Definition:

  - Quantitative data refers to numerical information that can be measured and counted. It represents quantities or amounts.

Examples:

- Age (20 years)
- Height (170 cm)
- Income (₹25,000)
- Test scores (85 out of 100)
- Number of laptops in a class (15)

9. What are the different types of data in statistics?

- 🔹 1. Qualitative Data (Categorical

Data) Definition:

- Data that describes qualities or characteristics, usually in words or labels.

a).

Nominal

Data-

Definition

:

- Categories without any natural order.
- Examples: Gender (Male/Female), Blood Group (A, B, AB, O), Eye color (Brown, Blue)

b).

Ordinal

Data

Definitio

n:

- Categories with a meaningful order, but no consistent difference between them.
    - Examples:Education level (High School < Bachelor's < Master's), Customer Satisfaction (Poor, Average, Good)

🔹 2. Quantitative Data (Numerical

Data) Definition:

- Data that represents quantities and can be measured or counted.

a). Discrete Data

- Definition: Countable numbers; values are separate and distinct.
- Examples: Number of students, Number of cars, Test scores (e.g., 90 out of 100)

b).

Continuous

Data

Definition:

- Can take any value within a range; often includes decimals.
- Examples: Height, Weight, Temperature, Time, Salary

10. Explain nominal, ordinal, interval, and ratio levels of measurement.

- 1.Nominal

Level Definition:

- Data is categorized into distinct groups or names without any order or ranking.

Characteristics:

- Categories are labels or names.
- No inherent numeric value or order.

Examples:

- Gender (Male, Female)
- Blood group (A, B, AB, O)

- Types of fruits (Apple, Banana, Orange)

2. Ordinal

Level

Definition:

- Data is categorized into groups that have a meaningful order or ranking, but the intervals between ranks are not necessarily equal.

Characteristics:

- Shows relative position.
- Differences between categories are not measurable or not equal.

Examples:

- Education level (High School < Bachelor < Master)
- Customer satisfaction ratings (Poor, Fair, Good, Excellent)
- Class ranks (1st, 2nd, 3rd)

3. Interval

Level

Definition:

- Numeric data with meaningful and equal intervals between values, but no true zero point (zero does not mean absence).

Characteristics:

- Can add and subtract values.
- Ratios are not meaningful (e.g., 20°C is not “twice as hot” as 10°C).

Examples:

- Temperature in Celsius or Fahrenheit
- IQ scores
- Calendar years (e.g., 2000, 2020)

4. Ratio Level

    - Definition: Numeric data with meaningful intervals and a true zero point, allowing for all arithmetic operations including meaningful ratios.

Characteristics:

- Has an absolute zero (absence of quantity).
- Can add, subtract, multiply, and divide.

Examples:

- Weight (0 kg means no weight)
- Height (0 cm means no height)
- Income (0 means no income)
- Age (0 years means just born)

11. What is the measure of central tendency?

- Measure of Central Tendency is a statistical concept that represents the center or typical value of a dataset. It provides a single value that summarizes the entire data by identifying the middle or average point.

12. Define mean, median, and mode.

1. Mean

-The mean (or average) is the sum of all the data values divided by the number of

values Example:

- For data 2, 4, 6, 8,
- Mean = (2 + 4 + 6 + 8) / 4 = 20 / 4 = 5

2. Median

- The median is the middle value of a dataset when the values are arranged in ascending or descending order. If there is an even number of observations, the median is the average of the two middle numbers.

Example:

- For data 3, 5, 7,
- Median = 5 (middle value)

For data 3, 5, 7, 9, Median = (5 + 7) / 2 = 6

3. Mode

- The mode is the value that occurs most frequently in the dataset. A dataset may have one mode, more than one mode, or no mode at all.

Example:

- For data 2, 4, 4, 6, 8,
- Mode = 4 (occurs twice, more than any other value)

13. What is the significance of the measure of central tendency?

1. Summarizes Data
    - It condenses large datasets into a single representative value, making it easier to understand and interpret.
2. Facilitates Comparison

- Allows comparison between different datasets or groups by comparing their central values.

3. Foundation for Further Analysis
    - Used as a basis for other statistical methods and tests, such as variance, standard deviation, and hypothesis testing.
4. Decision Making
    - Helps in making informed decisions in business, research, healthcare, etc., by understanding the “average” behavior or central value.
5. Identifies Trends and Patterns
    - Central tendency measures help detect trends in data, such as the average income, average score, or typical customer preference.

14. What is variance, and how is it calculated?

Variance is a measure of how much the data values spread out or deviate from the mean (average) of the dataset. It quantifies the degree of variation or dispersion in the data.

- A low variance means the data points are close to the mean.
- A high variance means the data points are spread out over a wider range.

📌 Steps to Calculate Variance:

- Find the mean of the data.
- Subtract the mean from each data value to get the deviation.

- Square each deviation.
- Find the average of these squared deviations.

15. What is standard deviation, and why is it important?

Standard deviation is a statistical measure that shows how spread out the values in a dataset are from the mean (average).

- It is the square root of variance, and it is expressed in the same units as the original data.

✅ Why is Standard Deviation Important:-

1. Measures Data Spread

- Tells you how consistent or varied the data points are.

2. Helps Understand Risk

- In finance or quality control, a high standard deviation indicates more risk or inconsistency.

3. Interprets Normal Distribution

- In a normal (bell-shaped) curve:
- ~68% of data falls within ±1 standard deviation
- ~95% within ±2
- ~99.7% within ±3

4. Improves Decision Making

- Useful in comparing the variability of two datasets (e.g., two investments, two test scores).

5. Foundation for Other Analysis

    - Important for advanced topics like confidence intervals, z-scores, regression, and hypothesis testing.

16. Define and explain the term range in statistics.

- Range is a basic measure of dispersion that shows the difference between the highest and lowest values in a dataset.

✅ Definition:

Range

Maximum Value − Minimum Value Range=Maximum Value−Minimum Value

17. What is the difference between variance and standard deviation?

Variance:

- Variance is the average of the squared differences between each data value and the mean.
    - It gives an idea of how much the values deviate from the average value, but the result is in squared units (not the same as the original data).
- It is useful in mathematical and theoretical analysis, especially in advanced statistics.

Standard Deviation:

- Standard deviation is the square root of the variance.
- It measures the average distance of each data point from the mean.
    - Unlike variance, standard deviation is expressed in the same unit as the original data, making it easier to interpret and more commonly used in practical applications.

Key Difference:

- Variance represents the squared dispersion, while standard deviation represents the actual average deviation.
- Standard deviation is generally preferred for interpretation because it is in the same unit as the data, whereas variance is primarily used in intermediate calculations.

18. What is skewness in a dataset?

- Skewness indicates whether the data distribution is symmetrical or tilted to one side. In a perfectly symmetrical distribution, the skewness is zero.

19. What does it mean if a dataset is positively or negatively skewed?

🔹 1. Positively Skewed (Right-Skewed) Distribution:-

- Tail is stretched to the right (higher values).
- Most data values are concentrated on the left (lower side).
- Mean > Median > Mode
- The mean is pulled up by a few high (extreme) values.

📌 Example:

- Income distribution in a population: most people earn average incomes, but a few earn extremely high salaries.

🔹 2. Negatively Skewed (Left-Skewed) Distribution:-

- Tail is stretched to the left (lower values).
- Most data values are concentrated on the right (higher side).
- Mean < Median < Mode
- The mean is pulled down by a few low (extreme) values.

📌 Example:

- Age at retirement: most people retire around 60–65, but a few retire much earlier (30s or 40s), pulling the average down.

20. Define and explain kurtosis.

- Kurtosis measures the “tailedness” of a data distribution. In simple terms, it indicates whether the data has more or fewer extreme values (outliers) than a normal distribution.

🔹 Types of Kurtosis:

1. Mesokurtic (Normal Kurtosis)

- Shape: Like a normal bell curve.
- Kurtosis value ≈ 3 (or 0 in excess kurtosis).
- Moderate tails, no extreme outliers.
- Example: Heights of people in a large population.

2. Leptokurtic (High Kurtosis)

- Shape: Tall and thin peak, fat tails.
- Kurtosis value > 3.
- More outliers; higher risk of extreme values.
- Example: Stock market returns.

3. Platykurtic (Low Kurtosis)

- Shape: Flat peak, thin tails.
- Kurtosis value < 3.
- Few or no outliers; data is evenly spread.
- Example: Uniform distribution.

21. What is the purpose of covariance?

🎯 Purpose of Covariance:

1. To Measure Direction of Relationship:

- Positive Covariance: Both variables move in the same direction.
- Negative Covariance: Variables move in opposite directions.
- Zero Covariance: No relationship between variables.

2. To Understand How Variables Co-vary:

- Helps identify whether changes in one variable are associated with changes in another.

3. Foundation for Correlation:

- Covariance is a base concept for correlation, which is a standardized version of covariance.

4. Used in Portfolio Theory (Finance):

    - Covariance helps measure the diversification benefit between assets (e.g., stocks) in investment.

22. What does correlation measure in statistics?

- Correlation shows how closely two variables move together. It is expressed by the correlation coefficient (r), which ranges from –1 to +1.

✅ Purpose of Correlation:

1. Identifies Relationships: Tells whether two variables move together (e.g., height and weight).
2. Helps in Prediction: Strong correlation can be useful for predictive models.
3. Simplifies Data Analysis: Detects associations that can guide deeper analysis.
4. Used in Regression Analysis: Correlation is a starting point before building regression models.

23. What is the difference between covariance and correlation?

✅ Covariance:

- Definition: Covariance measures the direction of the linear relationship between two variables.
- Indicates: Whether variables increase or decrease together.
- Range: Can take any real value (positive, negative, or zero).
- Units: The unit of covariance is the product of the units of the two variables, which makes it scale-dependent.
- Limitation: The magnitude of covariance is not standardized, so it's not useful for comparing relationships across different datasets.

✅ Correlation:

- Definition: Correlation measures both the strength and direction of the linear relationship between two variables.
- Indicates: How strongly related the variables are.
- Range: Always lies between –1 and +1.
- Units: It is a unitless (dimensionless) measure, because it is derived by dividing covariance by the product of standard deviations of the variables.
- Advantage: Correlation is standardized, making it suitable for comparing relationships across different datasets.

24. What are some real-world applications of statistics?

📘 Real-World Applications of Statistics

- Statistics plays a crucial role in many fields by helping people collect, analyze, interpret, and present data to make informed decisions. Here are some common real-world applications:

1. Healthcare and Medicine

- Designing and analyzing clinical trials for new drugs.
- Monitoring disease outbreaks and vaccine effectiveness.
- Evaluating patient outcomes and treatment methods.

2. Business and Marketing

- Analyzing customer behavior and preferences.
- Forecasting sales and market trends.
- Quality control and product testing.

3. Government and Public Policy

- Conducting censuses and surveys to gather population data.
- Evaluating the impact of policies and social programs.
- Crime statistics and public safety analysis.

4. Education

- Analyzing student performance and standardized test results.
- Improving teaching methods based on data.
- Resource allocation and policy planning.

Practical

1. How do you calculate the mean, median, and mode of a dataset?

- 📘 1. Mean (Arithmetic

Average) Steps:

1. Add all the values.
2. Divide by the total number of values.

- 📘 2. Median (Middle

Value) Steps:

1. Arrange the data in ascending order.
2. If the number of data points n n is odd, median = middle value.
3. If n n is even, median = average of the two middle values.

- 📘 3. Mode (Most Frequent Value)

Steps:

1. Count how many times each value appears.
2. The value with the highest frequency is the mode.

2. Write a Python program to compute the variance and standard deviation of a dataset.

In [ ]:

```
#2. Write a Python program to compute the variance and standard deviation
of a dataset.


# Dataset

data = [10, 12, 23, 23, 16, 23, 21, 16]


# Calculate mean

mean = sum(data) / len(data)


# Calculate variance

variance = sum((x - mean) ** 2 for x in data) / len(data)


# Calculate standard deviation

standard_deviation = variance ** 0.5


# Print results
print("Mean:", mean)
```

```
print("Variance:",
variance)

print("Standard Deviation:", standard_deviation)
```

```
Mean: 18.0

Variance: 24.0
Standard Deviation: 4.898979485566356
```

3. Create a dataset and classify it into nominal, ordinal, interval, and ratio types.

    - Here's a simple dataset with various types of data, and their classification into nominal, ordinal, interval, and ratio levels of measurement:

📊 Sample Dataset: Student Information

```
| Student ID | Gender | Grade Level | Satisfaction Rating |
Exam Score (out of 100) | Temperature (°C) |

|            |          |          |                      |

                       |                 |

| 001        | Male   | 12th       | Very Satisfied      | 88
| 36.5            |
| 002        | Female | 11th       | Satisfied           | 74
| 37.2            |
```

Student ID	Gender	Grade Level	Satisfaction Rating	Exam Score (out of 100)	Temperature (°C)
003	Male	10th	Neutral		35.8
004	Female	12th	Dissatisfied		

36.1 |

005 | Othe r | 11th | Very Dissatisfied
37.0 |

✅ Classification of Each Variable:

| **Variable** | **Type** | **Explanation**
|
| | | |

|
| **Student ID** | *Nominal* | Labels each student uniquely; no inherent order. |
| **Gender** | *Nominal* | Categorical label (Male/Female/Other); no ranking or order.
|
| **Grade Level** | *Ordinal* | 10th, 11th, 12th show a meaningful order, but not equal intervals.
|
| **Satisfaction Rating** | *Ordinal* | Ordered categories (Very Satisfied → Very Dissatisfied), but no fixed spacing. |
| **Exam Score** | *Ratio* | Has a true zero, and ratios make sense (80 is twice as much as 40). |

| **Temperature (°C)** | *Interval* | Ordered and has equal units, but

no true zero (0°C ≠ no temperature). |

In [ ]:

```
#4. Implement sampling techniques like random sampling and stratified sampling.

import pandas as pd


# Creating a sample dataset

data = {
    'StudentID': range(1, 21),
    'Gender': ['Male', 'Female']
    * 10,

    'Score': [65, 70, 80, 55, 90, 60, 75, 85, 67, 73, 88, 58, 62, 95, 79,
    68,

74, 77, 69, 81]
}

df =
pd.DataFrame(data)
print(df)
```

StudentID	Gender	Score

1	Male	65
2	Female	70
3	Male	80
4	Female	55
5	Male	90
6	Female	60
7	Male	75
8	Female	85
9	Male	67
10	Female	73
11	Male	88
12	Female	58
13	Male	62
14	Female	95
15	Male	79
16	Female	68
17	Male	74
18	Female	77

19	Male	69
20	Female	81

In

```
#5. Write a Python function to calculate the range of a dataset.
    def
        calculate_ra
        nge(data):
        if not data:

        return None # Handle empty list

    return max(data) - min(data)


# Example usage

dataset = [12, 5, 18, 7, 25, 10]
data_range = calculate_range(dataset)
print("Range of the dataset:", data_range)
```

```
Range of the dataset: 20
```

In [ ]:

```
#6. Create a dataset and plot its histogram to visualize skewness.

import numpy as np
import matplotlib.pyplot
as plt from scipy.stats
import skew

# Step 1: Create a right-skewed dataset
```

```
data = np.random.exponential(scale=2, size=1000)


# Step 2: Plot histogram

plt.figure(figsize=(8, 5))
plt.hist(data, bins=30, color='skyblue', edgecolor='black')
plt.title("Histogram of Skewed Data")
plt.xlabel("Valu
e")
plt.ylabel("Freq
uency")
plt.grid(True)
plt.show()


# Step 3: Measure skewness

skewness = skew(data)
print(f"Skewness of the dataset: {skewness:.2f}")
```

```
Skewness of the dataset: 1.91
```

In [ ]:

```
#7. Calculate skewness and kurtosis of a dataset using Python libraries.

import numpy as np

import pandas as pd
from scipy.stats import skew, kurtosis


# Create a sample dataset (e.g., exam scores)
data = [55, 62, 67, 70, 72, 75, 80, 85, 90, 95, 100, 105, 110]
```

```
# Using SciPy
skewness = skew(data)
kurt = kurtosis(data) # Default is Fisher's definition (normal dist = 0)


# Using Pandas (kurtosis returns Fisher's definition by default)
series =
pd.Series(data)
skewness_pd =
series.skew()
kurtosis_pd =
series.kurt()


# Print results
print("Using SciPy:")
print(f"Skewness:
{skewness:.2f}")
print(f"Kurtosis: {kurt:.2f}")


print("\nUsing Pandas:")
print(f"Skewness:
{skewness_pd:.2f}")
print(f"Kurtosis:
{kurtosis_pd:.2f}")
```

```
Using SciPy:

Skewness: 0.15
Kurtosis: -1.11


Using Pandas:

Skewness: 0.17
Kurtosis: -1.05
```

In [ ]:

```
#8. Generate a dataset and demonstrate positive and negative skewness.
```

```
import numpy as np
import matplotlib.pyplot
as plt from scipy.stats
import skew


# Generate positively skewed data (e.g., exponential distribution)

pos_skewed = np.random.exponential(scale=2.0, size=1000)


# Generate negatively skewed data (flip exponential)

neg_skewed = -np.random.exponential(scale=2.0, size=1000) + 10


# Calculate skewness
pos_skewness =
skew(pos_skewed)
neg_skewness =
skew(neg_skewed)

# Plot histograms
plt.figure(figsize=(12, 5))


# Positive skew

plt.subplot(1, 2, 1)
plt.hist(pos_skewed, bins=30, color='lightgreen',
edgecolor='black') plt.title(f"Positive Skew\nSkewness =
{pos_skewness:.2f}") plt.xlabel("Value")

plt.ylabel("Frequency")



# Negative skew

plt.subplot(1, 2, 2)
plt.hist(neg_skewed, bins=30, color='salmon', edgecolor='black')
plt.title(f"Negative Skew\nSkewness = {neg_skewness:.2f}")
plt.xlabel("Value")

plt.ylabel("Frequency")
```

```
plt.tight_layout()
plt.show()
```

In [ ]:

```
#9. Write a Python script to calculate covariance between two datasets.

    def
        calculate_covaria
        nce(x, y): if
        len(x) != len(y):

        raise ValueError("Datasets must have the same length.")



    n = len(x)

    mean_x =
    sum(x) / n
    mean_y =
    sum(y) / n
```

```
    covariance = sum((x[i] - mean_x) * (y[i] - mean_y) for i in range(n)) /
    (n
- 1)
    return covariance


# Example usage
dataset_x = [2, 4, 6, 8, 10]
dataset_y = [1, 3, 2, 5, 7]


cov = calculate_covariance(dataset_x, dataset_y)
print(f"Covariance between the datasets: {cov:.2f}")
```

```
Covariance between the datasets: 7.00
```

In [ ]:

```
#10. Write a Python script to calculate the correlation coefficient
between two datasets.

import numpy as np

from scipy.stats import pearsonr


# Sample datasets

x = [10, 20, 30, 40, 50]
y = [12, 24, 33, 45, 55]


# Manual calculation of Pearson correlation coefficient

    def
        manual_pearson_c
        orr(x, y): n =
        len(x)
    mean_x =
    sum(x) / n
```

```
    mean_y =
    sum(y) / n


    numerator = sum((xi - mean_x) * (yi - mean_y) for xi, yi in zip(x, y))
    denominator_x = sum((xi - mean_x)**2 for xi in x)

    denominator_y = sum((yi - mean_y)**2 for yi in y)



    denominator = (denominator_x * denominator_y) **
    0.5 return numerator / denominator


manual_corr = manual_pearson_corr(x, y)

print(f"Manual Pearson Correlation: {manual_corr:.4f}")


# Using numpy

np_corr = np.corrcoef(x, y)[0, 1]
print(f"Numpy Pearson Correlation: {np_corr:.4f}")


# Using scipy

scipy_corr, p_value = pearsonr(x, y)
print(f"Scipy Pearson Correlation: {scipy_corr:.4f} (p-value:
{p_value:.4f})")
```

```
Manual Pearson Correlation:
0.9992 Numpy Pearson
Correlation: 0.9992

Scipy Pearson Correlation: 0.9992 (p-value: 0.0000)
```

In [ ]:

```
#11. Create a scatter plot to visualize the relationship between two
variables.


import matplotlib.pyplot as plt
```

```
# Sample data

x = [5, 7, 8, 7, 2, 17, 2, 9, 4, 11]
y = [99, 86, 87, 88, 100, 86, 103, 87, 94, 78]


# Create scatter plot

plt.scatter(x, y, color='blue', marker='o')


# Add titles and labels

plt.title("Scatter Plot: Relationship between X and Y")
plt.xlabel("Variable X")

plt.ylabel("Variable Y")



# Show grid

plt.grid(True)


# Show plot

plt.show()
```

In [ ]:

```
#12.Implement and compare simple random sampling and systematic sampling.

import pandas as
pd import random


# Create a dataset of 20 items

    data =
        pd.DataFrame
        ({ 'ID':
        range(1,
        21),
```

```
'Value': [23, 45, 12, 67, 34, 89, 28, 76, 54, 31,

          38, 47, 59, 61, 72, 85, 90, 14, 55, 66]
  })


  def simple_random_sampling(df, n):

      # Randomly sample n rows without replacement
      return df.sample(n=n, random_state=42)


      def
          systematic_sampli
          ng(df, n): N =
          len(df)

      k = N // n # sampling interval

      start = random.randint(0, k - 1) # random start index


      indices = list(range(start, N, k))

      sampled_df =
      df.iloc[indices] return
      sampled_df


  # Number of samples to pick
  n_samples = 5


  # Perform sampling
  sample_srs = simple_random_sampling(data, n_samples)
  sample_sys = systematic_sampling(data, n_samples)


  print("Original Dataset:\n", data, "\n")
  print("Simple Random Sampling (n=5):\n", sample_srs, "\n")
  print("Systematic Sampling (n=5):\n", sample_sys)
```

		Original Dataset: ID Value
0	1	23
1	2	45
2	3	12
3	4	67
4	5	34
5	6	89
6	7	28
7	8	76
8	9	54
9	10	31
11	10	38
10	11	
11	11	47
11	12	
11	11	59
12	13	
11	11	61
13	14	
11	11	72
14	15	
11	11	85
15	16	
11	11	90
16	17	
11	11	14
17	18	

```
1   1      55
8   9
1   2      66
9   0
```

```
Simple Random Sampling (n=5):
```

```
      |    Val
      |    ue

0    1       2
             3
1    1       1
7    8       4
1    1       8
5    6       5
1    2       4
             5
8    9       5
             4
```

```
      Systematic
           Sampling
           (n=5): ID
           Value

3   4      67
7   8      76
11  12     47
15  16     85
19  20     66
```

In [ ]:

```
#13. Calculate the mean, median, and mode of grouped data.

import numpy as np


# Grouped data

class_intervals = [(10, 20), (20, 30), (30, 40), (40, 50), (50, 60)]
```

```
frequencies = [5, 8, 12, 20, 10]


# Calculate midpoints

midpoints = [(low + high) / 2 for (low, high) in class_intervals]


# Total frequency

N = sum(frequencies)


# Mean calculation

mean = sum(f * m for f, m in zip(frequencies, midpoints)) / N


# Cumulative frequency
cum_freq = np.cumsum(frequencies)


# Median calculation
median_class_index = np.where(cum_freq >= N/2)[0][0]
L = class_intervals[median_class_index][0] # lower boundary of median class
F = 0 if median_class_index == 0 else
cum_freq[median_class_index - 1] f_m =
frequencies[median_class_index]
h = class_intervals[0][1] - class_intervals[0][0] # class width
(assuming equal width)


median = L + ((N/2 - F) / f_m) * h


# Mode calculation

modal_class_index =
frequencies.index(max(frequencies)) L_modal =
class_intervals[modal_class_index][0]

f1 = frequencies[modal_class_index]

f0 = frequencies[modal_class_index - 1] if modal_class_index > 0 else 0
f2 = frequencies[modal_class_index + 1] if modal_class_index <
len(frequencies)
- 1 else 0
```

```
mode = L_modal + ((f1 - f0) / (2*f1 - f0 - f2)) * h


# Results

print(f"Mean: {mean:.2f}")

print(f"Median: {median:.2f}")
print(f"Mode: {mode:.2f}")
```

```
Mean: 39.00

Median: 41.25
Mode: 44.44
```

In [ ]:

```
#14. Simulate data using Python and calculate its central tendency
and dispersion.
import numpy as
np from scipy
import stats


# Simulate data - let's create a normal distribution with mean=50, std=10,
size=1000

data = np.random.normal(loc=50, scale=10, size=1000)


# Central Tendency
mean =
np.mean(data)
median =
np.median(data)
mode_result = stats.mode(data,
keepdims=True) mode = mode_result.mode[0]


# Dispersion
```

```
variance = np.var(data, ddof=1) # sample variance
(ddof=1) std_dev = np.std(data, ddof=1)  # sample
standard deviation data_range = np.ptp(data)        #
range = max - min


# Print results

print(f"Mean: {mean:.2f}")
print(f"Median: {median:.2f}")
print(f"Mode: {mode:.2f}")
print(f"Variance: {variance:.2f}")
print(f"Standard Deviation:
{std_dev:.2f}") print(f"Range:
{data_range:.2f}")
```

```
Mean: 50.02

Median: 49.52
Mode: 14.77
Variance: 97.39
Standard Deviation: 9.87
Range: 69.87
```

In [ ]:

```
#15. Use NumPy or pandas to summarize a dataset’s descriptive statistics.
import pandas as
pd import numpy
as np


# Create a sample dataset
data = {
    'Age': [23, 45, 31, 35, 62, 41, 29, 37, 55, 43],
    'Income': [50000, 64000, 58000, 60000, 72000, 65000, 52000, 61000,
    69000,
63000]
}


df = pd.DataFrame(data)
```

```
# Get descriptive statistics
summary =

df.describe()

print(summary)
```

	Age	Income
count	10.000000	10.000000
mean	40.100000	61400.000000
std	11.873874	6866.990284
min	23.000000	50000.000000
25%	32.000000	58500.000000
50%	39.000000	62000.000000
75%	44.500000	64750.000000
max	62.000000	72000.000000

In [19]:

```
#16. Plot a boxplot to understand the spread and identify outliers.

import matplotlib.pyplot
as plt import seaborn as
sns

import numpy as np


# Simulate a dataset with potential outliers
```

```
np.random.seed(42)
data = np.append(np.random.normal(50, 10, 100), [100, 105, 110]) # Adding
some outliers


# Plot using seaborn
plt.figure(figsize=(8, 5))
sns.boxplot(x=data,
color='skyblue')


# Add title and labels

plt.title("Boxplot to Visualize Spread and Outliers", fontsize=14)
plt.xlabel("Value", fontsize=12)

# Show
plot
plt.gri
d(True)
plt.sho
w()
```

In [20]:

```
#17. Calculate the interquartile range (IQR) of a dataset.

import numpy as np


# Sample dataset

data = [45, 50, 47, 52, 60, 53, 48, 56, 62, 75, 85]


# Calculate Q1 (25th percentile) and Q3 (75th percentile)

Q1 =
np.percentile(data,
```

```
25) Q3 =
np.percentile(data,
75)


# Calculate IQR

IQR = Q3 - Q1


# Print results

print(f"Q1 (25th percentile): {Q1}")

print(f"Q3 (75th percentile): {Q3}")
print(f"Interquartile Range (IQR): {IQR}")
```

```
Q1 (25th
percentile): 49.0
Q3 (75th
percentile): 61.0
```

Interquartile Range (IQR): 12.0

In [21]:

```
#18. Implement Z-score normalization and explain its significance.
import numpy as np

# Original dataset
data = np.array([10, 12, 15, 18, 20, 22, 25])

# Step 1: Calculate mean and standard deviation
mean = np.mean(data)
std_dev = np.std(data)

# Step 2: Apply Z-score normalization
z_scores = (data - mean) / std_dev
```

```
# Step 3: Print results print("Original
Data:", data) print("Mean:",
round(mean, 2))
print("Standard Deviation:", round(std_dev, 2)) print("Z-score
Normalized Data:", np.round(z_scores, 2))
```

Original Data: [10 12 15 18 20 22 25]
Mean: 17.43
Standard Deviation: 5.01
Z-score Normalized Data: [-1.48 -1.08 -0.48 0.11 0.51 0.91 1.51]

In [22]:

```
#19. Compare two datasets using their standard deviations.
import numpy as np

# Dataset A (less variation)
data_a = np.array([45, 47, 46, 48, 46, 45, 47, 46])

# Dataset B (more variation)
data_b = np.array([30, 50, 40, 60, 20, 70, 25, 65])

# Calculate standard deviation


std_a = np.std(data_a, ddof=1) # sample std dev
std_b = np.std(data_b, ddof=1)


# Print results
print("Dataset A:", data_a)
print("Standard Deviation of Dataset A:", round(std_a, 2))


print("\nDataset B:", data_b)
print("Standard Deviation of Dataset B:", round(std_b, 2))


# Interpretation
if std_a > std_b:
    print("\nDataset A is more spread out than Dataset
B.") elif std_b > std_a:
    print("\nDataset B is more spread out than Dataset
A.") else:
```

```
    print("\nBoth datasets have the same spread.")
```

```
Dataset A: [45 47 46 48 46 45  4  46]
                               :

Standard Deviation of Dataset  /  1.0
                               :  4

Dataset B: [30 50 40 60 20 70  2
                               5

Standard Deviation of Dataset  I  19.
                               :  09
```

```
Dataset B is more spread out than Dataset A.
```

In [23]:

```
#20. Write a Python program to visualize covariance using a heatmap.
import numpy as
np import pandas
as pd import
seaborn as sns

import matplotlib.pyplot as plt


# Step 1: Create a sample dataset
data = {
    'Height': [160, 165, 170, 175, 180, 185],
    'Weight': [55, 60, 65, 70, 75, 80],
    'Age': [22, 25, 30, 35, 40, 45]
}


df = pd.DataFrame(data)
```

```
# Step 2: Compute covariance matrix

cov_matrix = df.cov()


# Step 3: Plot heatmap

plt.figure(figsize=(8, 6))

sns.heatmap(cov_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Covariance Matrix Heatmap")

plt.show()
```

In [24]:

```
#21. Use seaborn to create a correlation matrix for a dataset.

import pandas as
pd import
seaborn as sns

import matplotlib.pyplot as plt
```

```
# Step 1: Create a sample dataset

data = {
    'Height': [160, 165, 170, 175, 180, 185],
    'Weight': [55, 60, 65, 70, 75, 80],
    'Age': [22, 25, 30, 35, 40, 45],
    'Score': [88, 85, 90, 92, 95, 89]

}


df = pd.DataFrame(data)


# Step 2: Compute correlation matrix
corr_matrix = df.corr()


# Step 3: Plot heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap='YlGnBu', fmt=".2f")
plt.title("Correlation Matrix Heatmap")

plt.show()
```

In [25]:

```
#22. Generate a dataset and implement both variance and standard deviation
computations.

import numpy as np


# Step 1: Generate a sample dataset
data = np.array([12, 15, 18, 21, 24, 27, 30])
```

```
# Step 2: Compute Mean
mean = np.mean(data)


# Step 3: Compute Variance (manual & NumPy)
manual_variance = sum((x - mean)**2 for x in data) / len(data) #
Population variance

numpy_variance = np.var(data)                                    # Also

population variance


# Step 4: Compute Standard Deviation (manual & NumPy)
manual_std_dev = manual_variance ** 0.5
numpy_std_dev = np.std(data)


# Step 5: Print the
results
print("Dataset:",
data) print("Mean:",
mean)
print("\nManual Variance:", round(manual_variance, 2))
print("NumPy Variance:", round(numpy_variance, 2))


print("\nManual Standard Deviation:", round(manual_std_dev,
2)) print("NumPy Standard Deviation:", round(numpy_std_dev,
2))
```

```
Dataset: [12 15 18 21 24 27 30]

Mean: 21.0


Manual Variance: 36.0

NumPy Variance: 36.0


Manual Standard
Deviation: 6.0 NumPy
Standard Deviation: 6.0
```

In [28]:

```
#23. Visualize skewness and kurtosis using Python libraries like matplotlib
or seaborn.

import numpy as
np import
seaborn as sns

import matplotlib.pyplot as plt

from scipy.stats import skew, kurtosis


# Step 1: Generate a dataset (positively skewed example)

data = np.random.exponential(scale=2, size=1000)

# Step 2: Calculate skewness and kurtosis
data_skewness = skew(data)
data_kurtosis = kurtosis(data) # Fisher's definition: normal = 0


# Step 3: Plot histogram + KDE
plt.figure(figsize=(10, 6))
sns.histplot(data, kde=True, bins=30, color='skyblue')
plt.title("Histogram with KDE: Skewness & Kurtosis
Visualization") plt.xlabel("Values")

plt.ylabel("Frequency")



# Step 4: Show skewness and kurtosis in plot
plt.axvline(np.mean(data), color='red', linestyle='--',
label='Mean') plt.legend()

plt.text(x=max(data)*0.5, y=100,

         s=f"Skewness: {data_skewness:.2f}\nKurtosis: {data_kurtosis:.2f}",
         fontsize=12, bbox=dict(facecolor='white', edgecolor='black'))


plt.show()
```

In [29]:

```
#24. Implement the Pearson and Spearman correlation coefficients for
a dataset.

import pandas as pd
from scipy.stats import pearsonr, spearmanr


# Sample dataset
data = {
    'X': [10, 20, 30, 40, 50, 60, 70],
    'Y': [15, 24, 33, 45, 52, 63, 72]
}
```

```
df = pd.DataFrame(data)


# Calculate Pearson correlation
pearson_corr, pearson_p = pearsonr(df['X'], df['Y'])


# Calculate Spearman correlation

spearman_corr, spearman_p = spearmanr(df['X'], df['Y'])


print(f"Pearson Correlation: {pearson_corr:.3f}, p-value: {pearson_p:.3f}")
print(f"Spearman Correlation: {spearman_corr:.3f}, p-value:
{spearman_p:.3f}")
```

```
Pearson Correlation: 0.999, p-value: 0.000

Spearman Correlation: 1.000, p-value: 0.000
```